Estructuras de Datos y Algoritmos

Grados en Ingeniería Informática

Examen Primer Cuatrimestre, 25 de Enero de 2018.

Nombre: _____ **Grupo:** _____

Laboratorio: _____ **Puesto:** _____ **Usuario de DOMjudge:** _____

Normas de realización del examen

1. Debes programar soluciones para cada uno de los tres ejercicios, probarlas y entregarlas en el juez automático accesible en la dirección `http://exacrc/domjudge/team`.

2. Escribe comentarios que expliquen tu solución, justifiquen por qué se ha hecho así y ayuden a entenderla. Calcula la complejidad de todas las funciones que implementes.

3. En el juez te identificarás con el nombre de usuario y contraseña que has recibido al comienzo del examen. El nombre de usuario y contraseña que has estado utilizando durante la evaluación continua **no** son válidos.

4. Escribe tu **nombre y apellidos** en un comentario en la primera línea de cada fichero que subas al juez.

5. Puedes descargar el fichero `http://exacrc/5781727.zip` que contiene material que puedes utilizar para la realización del examen (plantillas de código fuente y ficheros de texto con los casos de prueba de cada ejercicio del enunciado).

6. Tus soluciones serán evaluadas por el profesor independientemente del veredicto del juez automático. Para ello, el profesor tendrá en cuenta **exclusivamente** el último envío que hayas realizado de cada ejercicio.

1. (4 puntos) ¡Por fin ha nevado y han abierto las pistas de Valdesquí! Arya Voy está deseando probar la tabla de snow que le han traído los Reyes Magos. Esta temporada el mapa de pistas de Valdesquí ha cambiado de formato. Las zonas esquiables se han dividido en tramos y a los esquiadores se les suministra un listado de las alturas medias (sobre el nivel del mar) de esos tramos. Una secuencia de tramos *consecutivos* en el listado constituirá una **pista** solo si las alturas forman una secuencia decreciente (no necesariamente estricta). Arya valora más la longitud de una pista que su inclinación. Ayúdala a encontrar la pista más larga.

Especifica, diseña e implementa una función que reciba una secuencia de enteros positivos de longitud $0 \leq n \leq 1000$, representando las alturas de n tramos, y devuelva la longitud de la pista más larga. Escribe el *invariante* y la *función de cota* que permitan demostrar la corrección del algoritmo implementado. Por último, justifica el *coste* del algoritmo conseguido.

Entrada

La entrada comienza con una línea que contiene el número de casos de prueba. Cada caso de prueba contendrá el valor de n y a continuación las n alturas (enteros positivos menores que 4000) de los tramos.

Salida

Por cada caso de prueba el programa escribirá una línea con el número de tramos de la pista más larga (la que mayor número de tramos contiene).

Entrada de ejemplo

```
6
2
1000 1500
2
1500 1000
5
1500 1600 1200 1000 2000
6
1500 1200 1000 1000 1000 500
10
2250 2200 2300 2350 2325 2300 2210 2190 2101 1940
12
1000 950 900 825 910 800 750 700 650 650 500 425
```

Salida de ejemplo

```
1
2
3
6
7
8
```

2. (3 puntos) Dada una secuencia no vacía de n números enteros ordenados de manera no decreciente (puede haber elementos repetidos) y dado un entero x, se pide encontrar en la secuencia el entero *más cercano* a x (en términos de distancia, es decir, del valor absoluto de la diferencia). En caso de empate se debe devolver el menor. *Diseña e implementa* un algoritmo *recursivo* eficiente que resuelva el problema. Justifica el *coste* del algoritmo implementado.

Entrada

La entrada comienza con una línea que contiene el número de casos de prueba. Cada caso de prueba contendrá el valor de x, a continuación n y finalmente la secuencia de enteros.

Salida

Por cada caso de prueba el programa escribirá una línea con el entero más cercano a x contenido en la secuencia, según explicado en el enunciado.

Entrada de ejemplo

```
4
5
2
1 9
2
2
1 9
-2
7
-5 -3 -1 0 1 3 5
2
10
-1 0 1 1 2 2 2 3 6 9
```

Salida de ejemplo

```
1
1
-3
2
```

3. (3 puntos) Es época de nieves y a la CAM le preocupan los colapsos de tráfico en las carreteras de la Comunidad. Se dispone de m máquinas quitanieves para las $n(> m)$ carreteras. Pero no todas las máquinas pueden circular por todas las carreteras, pues cada máquina y cada carretera tienen una anchura y una máquina con anchura a solo podrá limpiar de nieve carreteras cuya anchura sea mayor o igual que a. Por otra parte, cada pareja máquina-carretera tiene asociada una calidad de limpieza. El objetivo es determinar, utilizando el esquema de *Vuelta atrás*, una asignación *válida* de máquinas quitanieves a carreteras que *maximice* la suma de las calidades de las carreteras que se han limpiado con las máquinas que se les ha asignado. Se ha de tener en cuenta que cada máquina puede limpiar a lo sumo una carretera y que cada carretera se puede limpiar a lo sumo con una máquina.

Entrada

La entrada comienza con una línea que contiene el número de casos de prueba. La entrada de cada caso de prueba consistirá en una primera línea con los valores de m y n, siendo $0 \leq m < n \leq 50$, una segunda línea con las anchuras de las m máquinas, una tercera línea con las anchuras de las n carreteras (todas las anchuras, de máquinas y carreteras, verifican $0 < a_i \leq 1000$), y finalmente m líneas de n números cada una que representan las calidades ($0 < c_{ij} \leq 1000$).

Salida

Por cada caso de prueba el programa escribirá una línea con la calidad máxima obtenida.

Entrada de ejemplo

```
3

1 2
2
1 2
7 5

2 3
2 3
4 1 5
20 1 15
15 1 5

3 6
2 3 6
2 2 5 4 8 9
10 25 6 12 20 8
5 14 10 10 13 9
16 16 17 12 11 5
```

Salida de ejemplo

```
5
30
46
```